# *Partie I*

# *État de l'art*

# Chapitre 1

# L'origine des hyperliens

**« Il est hélas devenu évident aujourd'hui que notre technologie a dépassé notre humanité »**

*Albert Einstein*

**Résumé du chapitre :** Les hyperliens sont considérés comme un concept clé dans la façon de conduire ou acheminer les connaissances du WWW. Dans certains secteurs tels que la « GED » et le « Knowledge Management », la gestion de liens est devenue une tâche complexe qui nécessite des techniques plus sophistiquées. Ce chapitre commence par survoler l'historique de l'hypertexte. Une définition formelle du lien est ensuite examinée en détail, ainsi que d'autres concepts tels que : liens typés, dynamiques, calculés, personnalisés et externes. Une attention toute particulière est accordée à l'accès et l'adressage des documents (Xlink et Xpath). Nous présentons également un panorama de la norme HyTime et ses fonctionnalités.

# Table des matières

# 1 L'origine des hyperliens et de l'hypertexte

L'hypertexte est un concept très ancien de gestion des écrits des bibliothèques (codex de parchemins, livres, encyclopédies, etc.) mais sa réalisation technique est récente. Dans son article [Rayward94], Rayward met en valeur les travaux du belge Paul Otlet sur les pratiques documentaires de la fin du XIX siècle jusqu'aux années 1930. Rayward décrit les réalisations d'Otlet et ses spéculations à propos des techniques de consultation de documents, ainsi que ses intuitions sur l'utilisation de la télévision comme moyen de communiquer l'information et ce qu'il nommait les «substituts de livres». Au vu de ses travaux, l'européen Otlet mérite autant d'être cité que l'américain Bush.

L'histoire contemporaine du concept d'hypertexte a débuté vers la fin de la seconde guerre mondiale et résulte des travaux de Vannevar Bush, Theodor Nelson et Douglas Englebart. Ces trois auteurs sont reconnus comme étant à l'origine du concept d'hypertexte lié à la technologie moderne. Depuis cette période, les termes préfixés par « *hyper* », pour définir des extensions du concept originel apparaissent de plus en plus souvent dans la littérature et dans les projets traitant de solutions aux problèmes de création, de gestion et de consultation d'information.

## 1.1 Vannervar Bush

La première description d'un système de type hypertexte a été proposée en 1945 par Vannevar Bush [Bush45] dans son article intitulé « *As We May Think* ». Bush a d'ailleurs été considéré comme le père du concept « *Hypertexte* » dans la communauté scientifique. Il considérait que les mécanismes d'indexation et de classification de l'époque ne pouvaient répondre à ses besoins. Dans son article il décrit ainsi son système Memex (**MEM**ory **EX**tender) de la façon suivante :

> *« Imaginons un futur système à usage personnel qui serait une sorte de fichier et une librairie privée automatique. Inventons-lui un nom : « Memex » fera l'affaire. Un Memex est un appareil dans lequel un individu peut stocker ses livres, ses données et ses communications, et qui est automatisé de manière à ce qu'il puisse les consulter avec*

*une rapidité et une flexibilité incomparable. C'est une extension à ses propres facultés mémorielles. »*

Memex repose sur une structure non linéaire de document, qui correspond aux capacités associatives du cerveau humain, où la possibilité est donnée d'explorer et d'annoter de l'information textuelle ou graphique. Compte tenu des moyens de restitution dont on disposait à l'époque (microfiches, papier, photos etc.) Bush n'a pu réaliser son projet. Du reste, il admettait que seuls d'importants progrès technologiques pouvaient rendre son projet réalisable.

Bush considère dans son approche que les connaissances se construisent par association (comme les pensées de l'être humain) et que les outils, par analogie, doivent appliquer ce même modèle de raisonnement.

## 1.2 Theodor Nelson

Theodor Nelson s'est lui-même défini comme un «Visionnaire de l'informatique » et s'est décrit comme un « Computopien » : un utopiste de l'ordinateur. Il est à l'origine, avec l'équipe de l'Université de Brown, de ce qu'il nomme « *Les hypertextes* ». De ce fait, il peut, c'est vrai, être considéré comme un des pionniers des systèmes hypertexte.

En 1965, il précise [Nelson65] que l'idée lui est venue en suivant un cours d'initiation à l'informatique, qui au début, devait l'aider à écrire ses livres de philosophie. Il défini le terme hypertexte pour spécifier des écrits non linéaires :

> *« Je cherchais un moyen de créer sans contrainte un document à partir d'un vaste ensemble d'idées de tous types, non structurées, non séquentielles, exprimées sur des supports aussi divers qu'un film, une bande magnétique présentant des portes derrière chacune desquelles un lecteur puisse découvrir encore beaucoup d'informations qui n'apparaissent pas immédiatement à la lecture de ce paragraphe. »*

Cette définition est très importante car elle va marquer toutes celles qui suivront cette époque. Il crée le projet Xanadu [Nelson95], [Nelson99], un projet hyper-

textuel et littéraire dont l'objectif est de créer une structure permettant de relier tous les ouvrages du monde à travers un réseau.

Un système hypertexte doit supporter des liens de n'importe quels types, conformément aux besoins de l'auteur. Les types de lien mentionnés dans [Nelson92] sont :

- des signets (bookmarks),
- des commentaires,
- des notes de bas de page,
- des sauts hypertexte.

Un autre type de lien introduit par Ted Nelson est le lien nommé « *Transclusion* » [Nelson95]. La « *Transclusion* » permet d'imbriquer logiquement tout ou partie de documents dans d'autres sans les inclure physiquement. Cette technique permet à un même document d'apparaître en de multiples endroits. On peut voir la transclusion comme un pointeur vers le document original depuis tous les documents qui le désignent ou qui sont liés à ce dernier. Cela évite la redondance de stockage.

## 1.3 Douglas Englebart

Douglas Englebart se démarque des deux précédents auteurs par le fait qu'il propose des outils, notamment dans le domaine des interfaces. Il est d'ailleurs à l'origine de la « *Souris* » qui est désormais présente sur tous les ordinateurs. En 1962, il a proposé « Augment », un système expérimental qui voulait être un outil d'amélioration des capacités intellectuelles de l'être humain [Englebart62]. C'était un environnement de travail en réseau qui proposait des outils de traitement de textes et de traitement d'idées et permettait la collecte d'informations en fournissant, entre autre, des mécanismes d'analyse et de résolution de problèmes.

L'évolution des techniques aidant, il présente dans les années soixante la mise en œuvre du concept de Vannervar Bush avec le premier système informatique en mode hypertexte : NLS – o**NL**ine **S**ystem. Ce système servait à archiver des articles, des notes, des bibliographies, etc. Tous les écrits liés à la recherche étaient enregistrés dans un système informatique permettant la mise en place de références

croisées entre des documents accessibles à tous. Ce système était assez proche d'une base de documents et facilitait le travail collaboratif [Englebart63].

# 2 Introduction aux hyperliens

La fonction principale de l'hypertexte ou plus généralement des hypermédia est en effet d'établir, dans un ensemble de documents, des possibilités de circulations « *Transverses* » c'est-à-dire, permettant d'ignorer à la fois la linéarité habituelle des documents et la distinction formelle entre documents. Une lecture hypertextuelle peut donc très bien sauter d'un passage à l'autre dans un document quelle que soit la distance physique entre ces passages. Ces « *Sauts* » sont rendus possibles grâce à la mise en œuvre de liens indiquant, à partir d'un endroit du document donné, qu'il est possible d'accéder à un autre endroit du même document ou d'un autre document. La figure 1-1 illustre cette fonction de navigation hypertexte [BLP+96].

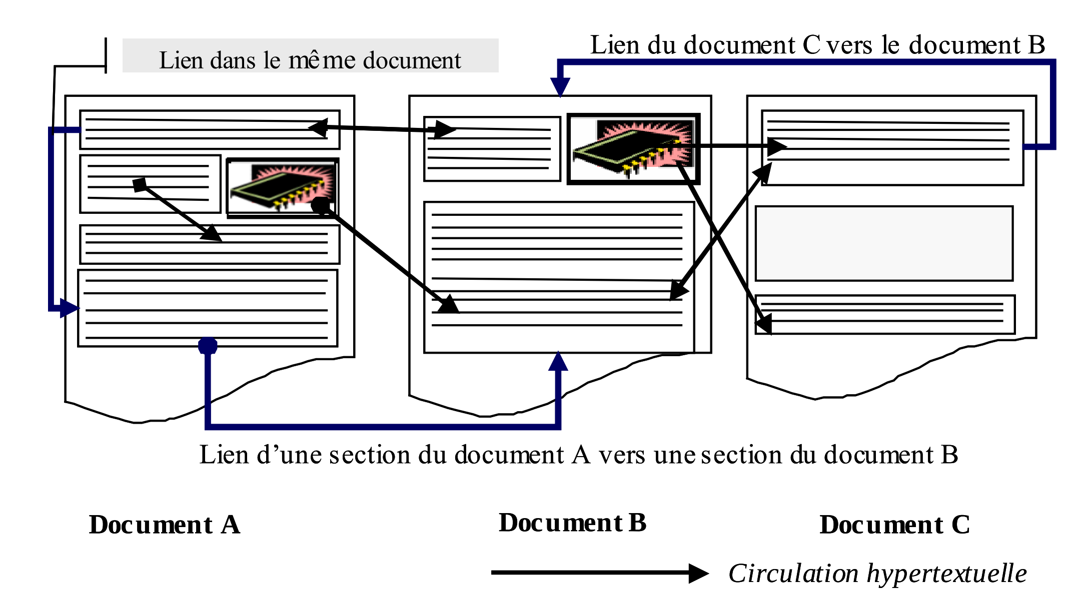



*Figure 1-1 Circulation hypertextuelle [BLP+96]*

D'après John B. Smith l'hypertexte est défini ainsi [SW88] :

> *« Une approche pour la gestion de l'information, où les données sont stockées dans un réseau des nœuds reliés (ou liés) par des liens. Les nœuds peuvent contenir du texte, des images, des documents audio et vidéo, aussi bien que d'autres formats de données. »*

Les hyperliens sont donc, avec les nœuds, une des notions fondamentales de l'hypertexte.

## 2.1 Nœuds

Les nœuds sont des unités élémentaires et indépendantes associées à des fragments d'information [TD96], le lecteur peut comprendre le contenu d'un nœud sans aucune connaissance du contenu d'autres nœuds. La notion de nœud constitue d'une certaine façon l'unité élémentaire de lecture. Dans les systèmes hypertexte, ils serviront de point d'arrivée de l'ancre. Ils peuvent inclure des contenus de type différent : texte, images, vidéo, audio. Des métadonnées[1] tels que la date, l'auteur, le langage et le nom du nœud peuvent leur être attachés.

Les nœuds d'information sont habituellement reliés entre eux et l'ensemble des liens unissant ces nœuds dessinent ce qu'il est convenu d'appeler des réseaux d'information. La figure 1-2 illustre un graphe des nœuds.

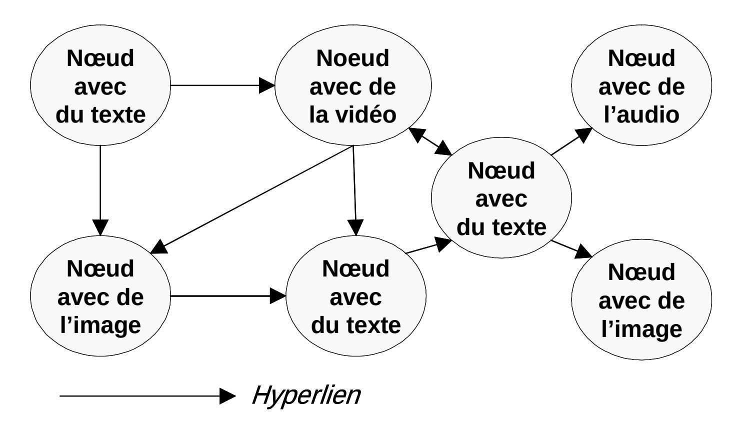



*Figure 1-2 Graphe des nœuds contenant des différents médias*

[1] Métadonnées : sont des données sur des données, par example, la taille d'un fichier est une métadonnée car c'est une donnée sure le fichier, et non un élément du fichier lui-même.

## 2.2 La norme HyTime

Cette section va nous permettre de présenter la norme HyTime, avec les principales fonctionnalités [Newcomb91], [Goldfarb93], [Newcomb95]. La norme HyTime s'appuie sur un ensemble de constructions SGML appelées formes architecturales et organisées en modules.

HyTime a été conçu de façon modulaire afin de faciliter son implémentation et son utilisation. Les formes architecturales sont classées en six modules qui définissent chacun une fonctionnalité précise. Les relations de dépendance entre les modules sont résumées par la figure ci-dessous.

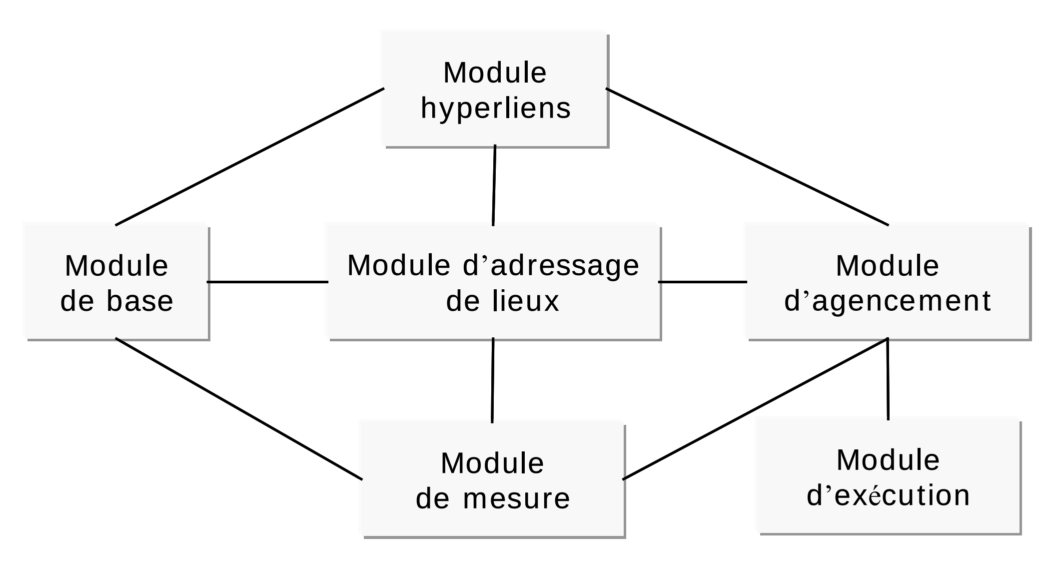



*Figure 1-3 Les modules de HyTime*

### 2.2.1 Le module de base

Le module de base est le composant obligatoire et principal de l'architecture HyTime [Afnor94]. Il est composé de fonctionnalités indépendantes dont certaines sont optionnelles. Les fonctionnalités obligatoires concernent les fonctionnalités de gestion de l'hyperdocument, c'est-à-dire la représentation, l'identification et l'accès aux objets, ainsi qu'un format d'échange d'hyperdocument.

### 2.2.2 Le module de mesure

Ce module contient les fonctionnalités de repérage par coordonnées. Il est optionnel mais constitue un prérequis pour toute utilisation de lieux repérés par leurs coordonnées dans le module d'adressage de lieux, ainsi que pour le module d'agencement. La fonctionnalité principale de ce module est l'adressage par coordonnées. Il s'agit de spécifier un lieu par la position relative d'un objet.

### 2.2.3 Le module d'adressage de lieux

Les plus simple moyen de localiser un élément est lui associer un attribut ID. Cette forme de localisation est mise en œuvre par SGML et par le module de base de HyTime. Cependant le module de base ne fournit aucun moyen d'affecter un ID à des portions arbitraires de données, ni à des collections, ni à des éléments ne pouvant avoir d'ID.

### 2.2.4 Le module hyperliens

Le module hyperlien permet d'effectuer des connexions ou "hyperliens" entre objets, soit à l'intérieur d'un document unique, soit parmi les documents et les objets informationnels qui constituent un hyperdocument.

### 2.2.5 Le module d'agencement

Ce module permet d'agencer les événements ou occurrences d'objets sur des axes finis de coordonnés, ces coordonnées pouvant être exprimées selon des unités spatiales ou temporelles. Ce mécanisme permettant également de repérer les événements les uns par rapport aux autres.

### 2.2.6 Le module d'exécution

Quand le module d'agencement est utilisé on pet contrôler l'exécution de l'application en terme de modification d'objets (ordre de modification, modificateurs autorisés, etc) et en terme de projection d'événements sur les axes espace/temps (durée d'un slide, fréquence d'un son, taille d'une image, etc.).

Après avoir décrit les fonctionnalités de HyTime, nous allons donc survoler l'utilisation de HyTime pour la modélisation de liens. Ce qui fait la force des fonc-

tionnalités hyperliens de HyTime est l'indépendance qu'il procure vis-à-vis des systèmes pour exprimer des adresses des objets hypermédias. HyTime fait une distinction entre les liens et les extrémités de liens ou ancres.

La granularité de HyTime en ce qui concernel'adressage permet de définir comme ancre : un document, un élément d'une instance, quelques mots et jusqu'à une partie d'image (grâce aux hospots). Une ancre potentielle est tout ce qui peut être adressé ou identifié par HyTime.

Les hyperliens sont des fonctionnalités du module hyperlien. Les lieux et leur adressage sont définis dans le module d'adressage de lieux. Le module d'adressage de lieux utilise également le module de mesure pour l'adressage par coordonnées. Le module de mesure permet de spécifier la position et l'étendue des objets dans différentes unités de mesure définis par l'application. Les modules utilisés pour la modélisation des liens sont donc les modules hyperlien, adressage de lieu et mesure.

## 2.3 Les hyperliens

Un des concepts essentiels des systèmes hypertexte/hypermédia sont les concept de lien et hyperlien. Ils sont à la base de la structure d'un document hypertexte/hypermédia respectivement. De façon simple, un lien relie un nœud source à un ou plusieurs nœuds destination. Les extrémités des liens sont appelées « *ancres* » et décrivent l'attachement d'un lien vers un nœud. Le point de départ du lien correspond à l'ancre source, et la destination du lien correspond à l'ancre cible [Maurer96].

Les ancres peuvent être attachées soit à un nœud entier, soit à une région d'un nœud (une section, un paragraphe, etc), soit à un point dans un nœud (un mot). La direction ou sens d'un lien indique si le lien est unidirectionnel ou bidirectionnel. Pour le lien unidirectionnel, la navigation se fait dans un seul sens de la source vers la cible, tandis que les liens bidirectionnels permettent une navigation dans les deux sens de la cible vers la source et vice-versa. La figure 1-4 illustre ces cas.

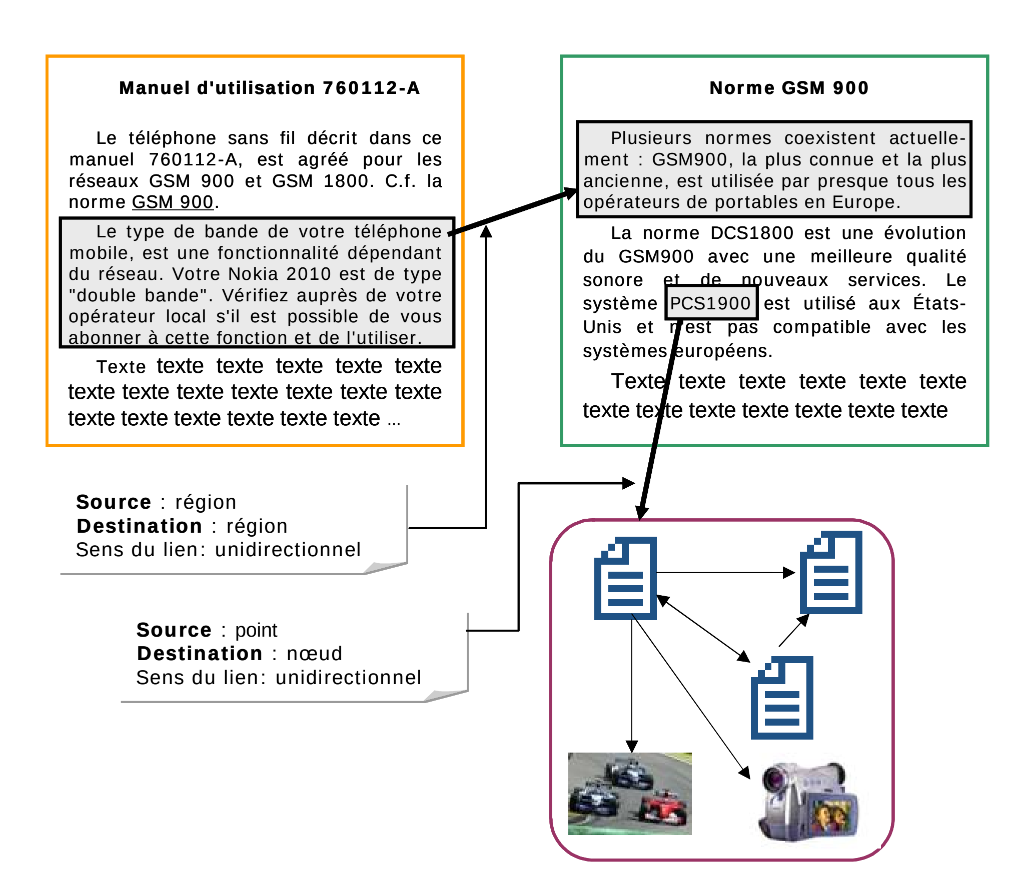



*Figure 1-4 Réseau hypertexte avec différents types d'ancres*

La structure constituée par des liens peut être vue de manière abstraite comme un graphe avec des sous-graphes de différents types, par exemple un arbre, un graphe acyclique ou un graphe dirigé. Derrière la structure constituée par les liens, chaque lien peut avoir son propre type. (C.f. section 2.3).

Cette section présente une définition formelle des liens réalisée par Liam Quin [Qui98].

**Définition 1.1**

> « *Un lien est une assertion, où une relation R est établi entre deux ensembles de ressources* »

**Définition 1.2**

Un lien peut aussi être défini comme une fonction : $f_R : S_s \rightarrow S_t$

Où $R$ est la relation d'assertion et $S_s$ et $S_t$ (le domaine et la fonction de lien respectivement). La relation doit exister entre ces deux ensembles de ressources.

**Définition 1.3**

Un lien peu également défini comme une fonction inverse :

La fonction du lien inverse $f^{-1}$ est la fonction qui établi le sous-domaine $S_t$ de $f$ derrière le domaine $S_s$.

À savoir deux types de fonctions inverses : calculée et explicite.

**Définition 1.4**

Lien comme fonction inverse calculée :

La fonction calculée inverse $f$-1 de la fonction $f$R : Ss → St est la fonction obtenue par l'évaluation de f, donc, on commence par $f$-1 est St → Ss. Un exemple de la fonction inverse calculée, pourrait être la navigation par le bouton précèdent via l'intermédiaire du historique stocké dans « une sorte de cache ». Les browsers web sont des applications basées sur cette technique (normalement avec l'aide de bouton précédent).

**Définition 1.5**

Lien comme fonction inverse explicite :

La fonction inverse explicite $f^{-1}$ de la fonction : $f_R$ : $S_1 \rightarrow S_2$ est une fonction $f^3$: $S_s \rightarrow S_4$, avec les contraintes $S_s \rightarrow S_3$ identiques ou avec une intersection non vide *P*. *P* peut être un sous-ensemble vrai de $S_2$ depuis il établi l'inverse et ne peut pas être défini par tous les membres de l'ensemble du résultat de f.

De façon générale, les fonctions de lien ne sont pas « injectives » parce que la relation de l'assertion peut avoir des valeurs : (1 à-1), (n à 1), (n à n ), c'est-à-dire que la définition formelle présenté dans cette section n'impose aucune restriction sur la cardinalité des ensembles de ressources. Un lien entendu, peut avoir plusieurs cibles (locales ou distantes).

Les fonctions de lien ne sont pas nécessairement surjectives parce qu'il existe des ressources dans un co-domaine de la fonction du lien (par exemple toutes les ressources accessibles dans le World Wide Web) qui ne peuvent pas être une cible d'un lien. La fonction de lien ne peut pas non plus être non déterministe, c'est-à-dire, à chaque fois que la fonction est évaluée elle renvoie un différent ensemble des ressources.

En théorie, le calcul de la fonction inverse est toujours possible, mais pas dans la pratique, il peut devenir coûteux, voir impossible. La conclusion de cette fonction inverse est équivalente à la question suivante :

> *« Quel est l'ensemble des documents du World Wide Web qui pointent vers une page en particulière ».*

## 2.4 Type de liens

### 2.4.1 Liens externes

Ce type de liens concernent les liens vers des bases externes (cette technique correspond à l'approche « *Hyperbases* » (C.f. chapitre 2). Il y a une séparation nette entre le lien et le document vers lequel il pointe. Les avantages d'une telle base de lien « *Link Database* » sont les suivants :

- La facilité d'implémentation des liens bidirectionnels, et la maintenance aisée qui en découle.
- La modification du lien est indépendante du document.
- La création des nouveaux types de liens dans la base est autorisée. (C.f. Section 2.3.2)
- Le contrôle des droits d'accès aux liens est plus facile.

La création et maintenance des bases de données de liens externes nécessite un travail additionnel : par exemple l'implémentation d'un mécanisme pour garantir la cohérence et/ou l'intégrité des liens [Kappe95a]. Ceci est au cœur de nôtre problématique de recherche. Parmi quelques systèmes hypertexte qui mettent en œuvre ce type de mécanisme, on peut citer : HyperWave [Maurer96], Microcosm par Davis [DHH+92],[Davis95, 98],[OHS98],[FHH+90] Dexter par Halaz [HS90, HS94].

### 2.4.2 Liens personnalisés

Un lien personnalisé permet de mettre en relation le lien et un certain niveau de connaissance. Si nous observons la structure de la figure 1-5, et supposons qu'elle représente tous les documents d'une base de données, nous constatons que chaque

document comporte un grand nombre de références vers des documents parfois similaires, mais ne présentent pas tous un intérêt pour un utilisateur donné. La base de données est pourtant présentée de la même façon à tous.

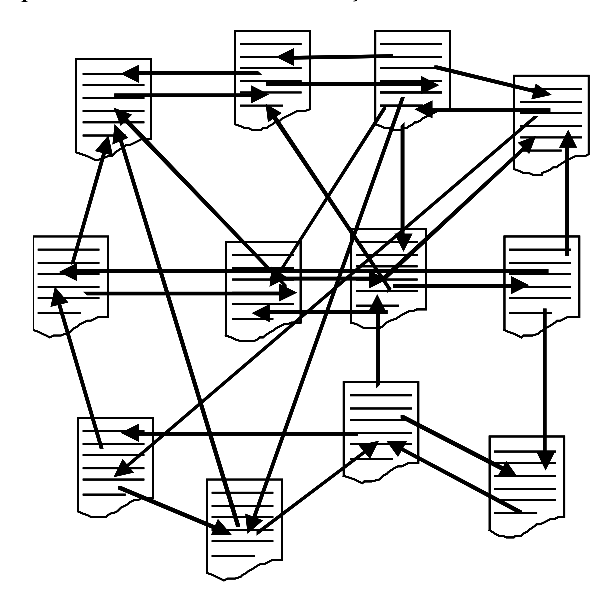


*Figure 1-5 Réseaux des liens non filtrés*

La personnalisation de liens va permettre à l'utilisateur de « filtrer » les liens pour sa requête. Ce n'est alors que les documents référencés qui seront accessibles par le lien personnalisé. La figure 1-6 reprend la figure 1-5, avec une personnalisation.

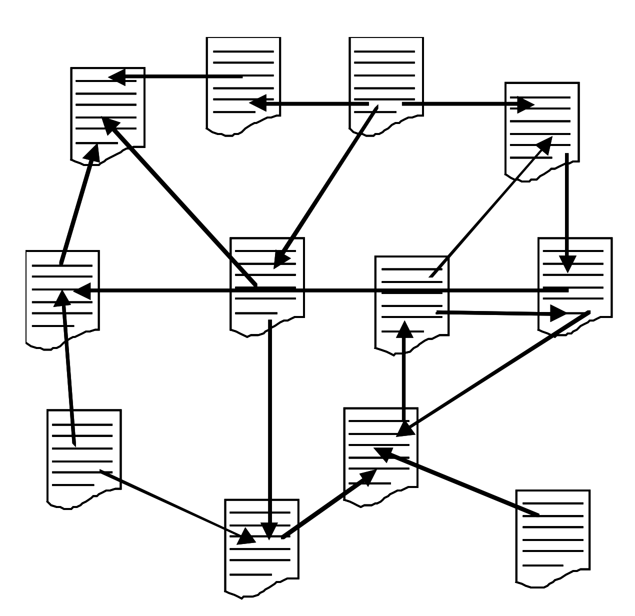


*Figure 1-6 Réseaux des liens personnalisés*

La personnalisation peut être réalisée de deux manières différentes :

1. Les utilisateurs décident eux-mêmes quels liens seront présentés.
2. Le système hypertexte réalise un calcul des liens par rapport aux préférences, aux connaissances, au degré d'expertise ou autres caractéristiques de l'utilisateur.

Les systèmes qui effectuent cette sorte d'adaptation des liens ou contenus sont nommés Systèmes Hypertextes Adaptatifs (*Adaptive Hypermedia System AHS*), [DGH99], [BSW96]. La personnalisation des pages n'avait pas été considérée lors de la conception initiale du Web. Ce sont les sites Web commerciaux qui ont été poussés, par la force du commerce électronique, à rendre le World Wide Web plus personnalisable [Goodman98].

De nos jours, certains sites Web permettent aux utilisateurs de personnaliser les pages qu'ils visitent de manière dynamique, en fonction de leurs préférences ou de leurs habitudes de navigation. Les techniques actuelles qui permettent de rendre le contenu d'un site Web plus dynamique sont les Scripts-CGI (*Common Gateway Interface*) [CR98], les pages ASP (*Active Server Pages*) [Microsoft02] et les Servlets de Java [Hunter99]. Toutes ces techniques fonctionnent d'une manière similaire : touts les pages sont traitées du côté serveur, et seules les pages pertinents sont envoyées au client.

Les données sont stockées sur le serveur Web, une authentification est réclamée à l'utilisateur pour l'identifier, parfois elle est accolée au client sous forme de « cookies». Une combinaison des deux peut être utilisée (les données d'authentification sont stockées sur des « cookies », et les données de personnalisation sur le client). Le plus souvent, les serveurs Web ne personnalisent que le contenu des pages, tandis que la structure du site et de ses liens reste la même. Ceci est liée a une limitation d'HTML : il est quasiment impossible de présenter différentes structures et liens pour un même document étant donné que les liens sont inclus dans les documents eux-mêmes.

### 2.4.3 Liens typés

Sémantiquement, les liens typés aident les auteurs à organiser leur information de façon plus efficace et fournissent aux utilisateurs des informations pertinentes par rapport à l'objectif du lien en cours d'exploration. Dans le domaine de la recherche d'information, cela facilite l'automatisation et constitue un avantage non négligeable pour les moteurs de recherche [ASR94]. Le système ParaSite [Spertus97] distingue quatre types de liens (ascendant, descendant, croisé et externe).

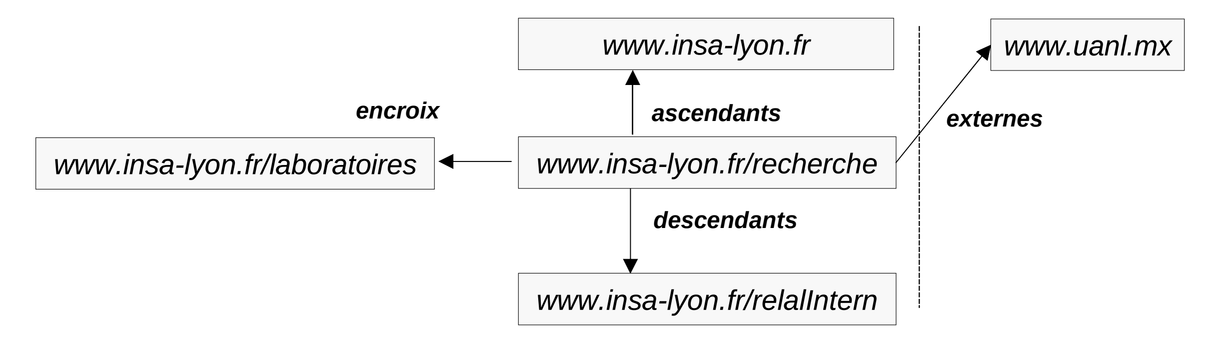



*Figure 1-7 Représentation des liens hypertextuels dans le système ParaSite [Spertus97]*

Les liens typés sont également abordés par Wang et Rada [WR98] qui de plus décrivent le système hypertexte comme un réseau sémantique : un réseau sémantique peut prendre la forme d'un graphe, dans lequel les concepts sont des nœuds; et les relations entre les nœuds sont des liens. Ils distinguent trois types de systèmes hypertextes basés sur ce paradigme dit «sémantique» :

- *Un système hypertexte non-structuré* admet des nœuds et liens de type arbitraires, leur utilisation n'impose pas de contraintes. La sécurité du système dépend totalement des actions de chaque auteur et/ou utilisateur.
- *Un système hypertexte semi-structuré* suggère un ensemble de types de liens mis à disposition de l'auteur et/ou utilisateur.
- *Un système hypertexte structuré* inclut des règles afin de définir les contraintes de type structurelle et relationnelle, ces contraintes étant imposées par le système hypertexte lui-même.

Le World Wide Web peut être considéré comme un système hypertexte semi-structuré. En 1996 Maloney et *al* [RCBM+97] a proposé une extension à la spécification d'HTML version 2.0 [BC90] au niveau des balises < `a` > et < `link` >. Ces modifications étant minimes, sa proposition a été acceptée et est devenue une recommandation du consortium W3C [W3C99]. Conformément à la recommandation de ce consortium, les deux balises < `a` > et < `link` > détiennent un attribut **rel** et un attribut **rev**, qui indiquent respectivement les fonctions suivant et précédent.

La version initiale de la spécification d'HTML permet d'énumérer différentes catégories de types de liens (attention, quelques uns sont différents de ceux que nous avons décrits dans la section 2.3) :

1. **liens définis dans les navigateurs (bouton home, précédent et suivant),** ces liens sont utilisés par des agents spécifiques (clients Web ou Navigateurs) pour accéder à l'historique.
2. **liens de navigation à travers les nœuds (contenu, index, navigateur)** consiste à aider les utilisateurs en leur proposant une table des matières ou une liste de documents relatifs à leurs recherches, en indiquant leur localisation dans une hiérarchie.
3. **liens de hiérarchie (père ou nœud racine, fils, etc.)** ces liens peuvent être utilisés par un programme ou un agent de façon transparente à l'utilisateur dans le but de visualiser la structure hiérarchique des documents.
4. **liens de séquence (démarrer, finir, suivant, précédent)** proposent à l'auteur du document la possibilité d'indiquer aux utilisateurs une séquence qu'ils devraient suivre.
5. **liens avec des métadonnées.** Pour les informations qui ne sont pas constitutives du document, il est utile d'ajouter ces **métadonnées** pour les associer de façon claire et non ambiguë au document. Nous pouvons considérer l'auteur, l'éditeur, le copyright, le rédacteur comme des métadonnées.

Nous décrivons les différents types de liens proposés par la recommandation du consortium W3C. Malheureusement, les navigateurs (*Netscape Communicator et Internet Explorer*) ne sont pas prêts à supporter cette spécification de types de liens.

Une des raisons est sans doute que la spécification du W3C ne fait aucune suggestion pour l'implémentation [Goodman98].

Le langage XLink [W3C-XLink], [Laurent99] offre plus de possibilités pour intégrer des métadonnées aux hyperliens. Toujours candidat à la recommandation W3C, XLink deviendra probablement une nouvelle norme, un nouveau standard dès que les navigateurs implémenteront toutes les fonctionnalités de XML [W3C-XML], [Ray01],[HN01],[ McLaughlin02b],[Laurent99]. (C.f. Section 3) pour approfondir les capacités de Xlink.

### 2.4.4 Liens dynamiques

Une des nombreuses applications basée sur le Web concernée par les fonctionnalités de liens sont les librairies numériques (DL - Digital librairies). Leur principe consiste à récupérer le contenu d'une base de données et à générer tous les liens nécessaires à la volée. Dans la plupart des cas, cela exige des auteurs l'établissement lors de l'écriture des documents d'un schéma prédéfini. L'information de la structure de liens est une propriété du programme ou un « fichier script » qui génère la page finale. Le calcul selon une règle prédéfinie des ancres source et cible du lien est à la base des liens dynamiques [AV94], [AVC94].

Les liens dynamiques sont générés à partir d'une fonction implicite pour les liens. Le lien déclaré est un lien du même type que le lien dynamique. Avec ce type de lien, le calcul des extrémités est spécifié au moment de la création, il est réalisé au moment du franchissement.

De nos jours, nous voyons par exemple que la publication des revues sur le Web est devenue une chose appréciée du monde de la recherche. Plusieurs efforts ont été faits pour publier des revues sur le Web. La technique la plus employée a consisté à laisser le contenu intact et à permette une navigation simple. Des techniques plus sophistiquées ont été employées : usage des Scripts-CGI [CR98], pages ASP [Microsoft02] et des Servlets de Java [Flanagan2000] afin d'imposer une structure définie de liens [CH98]. Un exemple explicite est l'énorme librairie numérique d'ACM, elle propose des liens entre les articles du même sujet et tous les articles publiés par le même auteur.

# 3 XML Linking Language (XLL)

XLL est un langage de description des liens hypertextes pour XML. XLL faisant encore aujourd'hui l'objet d'études par le W3C, les spécifications évoluent et ne sont pas encore parfaitement opérationnelles au sein de toutes les applications XML.
Trois spécifications constituent le cœur du langue XLL : XLink, XPath et XPointer.

- XPath et XPointer indiquent comment des éléments et des données sont adressés dans un document XML.
- Xlink, spécifie comment des liens doivent être imbriqués dans des documents XML. XLink permet d'accéder à un document par l'intermédiaire d'un URI « *Uniform Resource Identifier* » à l'image de la balise <a href="URL"> du langage HTML.

Dans les sections suivantes, les trois modules du XLL sont abordés : leurs capacités, usage et quelques problèmes de spécifications.

## 3.1 XLink : le langage de liaison de XML

La norme Xlink est une recommandation du W3C parue en juin 2001 [W3C-XLink]. XLink propose, dans la lignée de XML, une façon normalisée de représenter des liens hypertextuels plus riches que ceux permis par HTML. XLink n'est actuellement qu'un projet de norme du Consortium W3. L'autre exemple, le formalisme des URI (*Uniform Resource Identifier*), est en cours d'élaboration par le W3C; ce concept chapeaute la syntaxe URL (*Uniform Resource Locator*) [rfc1738] et URN *(Uniform Resource Name*) [rfc1737]. Actuellement, un des seuls cadres de localisation, un tant soit peu normalisé et réellement implanté, est le formalisme des URL, sur lequel sont basés les liens hypertextuels de HTML

Cette norme laisse entrevoir des possibilités importantes quant à son utilisation dans le cadre de la gestion documentaire. En effet, il apparaît possible de rajouter des annotations à un document initial sans que celui-ci soit modifié.

## 3.2 Possibilités de XLink

D'après le groupe de travail Xlink Working Draft [W3C-XLink].

> *« Xlink est un langage qui permet l'insertion d'éléments dans les documents XML afin de créer et de décrire des liens entre des ressources. »*

Dans le contexte XLink, un lien est une relation explicite entre deux ou plusieurs ressources ou parties de ressources. XLink permet :

- d'affirmer une relation de lien entre plus de deux ressources.
- d'associer des métadonnées à un lien
- de définir des liens unidirectionnels et bidirectionnels.
- d'attacher des liens aux documents en lecture seule.
- d'exprimer des liens qui résident dans une localisation séparée des ressources reliées. Par exemple dans un document XML séparé ou dans une base de données. (C.f. Section 2.3.1).

XLink est conçu pour préserver la simplicité du mécanisme de liaison de HTML, mais il fournit davantage de puissance et d'extensibilité si nécessaire. Il permet de créer des liens entre plus de deux ressources et d'associer des métadonnées aux liens. XLink définit deux sortes de liens fondamentaux : **liens simples et liens étendus**. La notion de liens étendus permet d'associer des ressources sans que celles-ci soient modifiées [Michard01]. Cette notion repose sur l'idée de document pivot : ce document servira de lien entre toutes les ressources.

## 3.3 XPath

XPath est le résultat d'un effort d'homogénéisation de la syntaxe et de la sémantique de fonctions communes à XSLT et XPointer. L'objectif premier de XPath est de définir la manière d'adresser des parties d'un document XML. En vue de pourvoir à cet objectif premier, cette spécification fournit également les moyens pour traiter des chaînes de caractères, des nombres et des booléens. XPath utilise une syntaxe compacte ne suivant pas la XML pour faciliter son utilisation dans des URI et des attri-

buts de balises XML. XPath agit sur les structures abstraites et logiques d'un document XML, plutôt que sur sa syntaxe apparente. [W3C-Xpath].

Le nom XPath vient de l'utilisation d'une écriture de type «chemins d'accès», comme les URL, pour se déplacer à l'intérieur de la structure hiérarchique d'un document XML [Ray01],[HN01],[ McLaughlin02b],[Laurent99].

- XPath est conçu pour être utilisé à la fois par XSLT et XPointer.
- XPath fournit un mécanisme pour associer une valeur de type chaîne de caractères à chaque type de nœud. Certains ont également un nom.
- XPath supporte complètement les espaces de nom XML).

XPath représente les documents XML comme un arbre de nœuds. Il y a plusieurs types de nœuds, parmi lesquels les nœuds d'éléments, d'attributs et de texte (en représentant les sections CDATA d'un document XML).

La forme syntaxique de base de XPath est l'expression. Une expression s'applique à quatre types possibles d'objets :

- ensemble de nœuds (une suite non-ordonnée de nœuds sans doublon)
- booléen (vrai ou faux)
- nombre (un réel)
- chaîne de caractères (une suite de caractères UCS)

L'évaluation des expressions dépend du contexte. XSLT et XPointer spécifient chacune le contexte pour les expressions XPath qui y sont utilisées. Le contexte peut être :

- un nœud (le nœud contextuel)
- une paire d'entiers positifs non nuls (la position contextuelle et la dimension contextuelle)
- un ensemble de variables associées
- une bibliothèque de fonctions

### 3.3.1 Chemins de localisation

La grammaire spécifiée dans cette section est compatible avec celle applicable aux valeurs d'attributs telle que définie à partir de la version 1.0 de la norme XML [W3C-Xml].

Une expression particulièrement importante est le chemin de localisation. Celle-ci sélectionne un ensemble de nœuds relativement à un nœud de contexte dit *nœud contextuel*. Le résultat de l'évaluation d'une telle expression est l'ensemble des nœuds ciblés par le chemin de localisation. Les chemins de localisation peuvent contenir des expressions récursives de filtrage des nœuds. Un chemin de localisation doit être conforme à la règle de production des chemins de localisation (C.f. Section 3.3.2).

Chaque chemin de localisation peut être exprimé en utilisant une syntaxe directe mais plutôt verbeuse. Il existe également un certain nombre d'abréviations syntaxiques permettant d'exprimer de manière concise les cas les plus courants. Cette section explique la sémantique des chemins de localisation en utilisant une syntaxe non-abrégée.

Ci-après sont présentés quelques exemples de chemins de localisation utilisant une syntaxe non-abrégée [W3C-Xpath] :

- `child::`**`section`** sélectionne l'élément **`section`** enfant du nœud contextuel ;
- `child::*` sélectionne tous les éléments enfant du nœud contextuel ;
- `child::text()` sélectionne tous les nœuds textuels du nœud contextuel ;
- `child::node()` sélectionne tous les enfants du nœud contextuel, quelque soit leur type ;
- `attribute::`**`prix`** sélectionne l'attribut **`prix`** du nœud contextuel ;
- `attribute::*` sélectionne tous les attributs du nœud contextuel ;
- `descendant::`**`section`** sélectionne tous les descendants **`section`** du nœud contextuel ;
- `ancestor::`**`section`** sélectionne tous les ancêtres **`section`** du nœud contextuel ;

- ancestor-or-self::**section** sélectionne tous les ancêtres **section** du nœud contextuel et le nœud contextuel lui-même si c'est une **section**;
- descendant-or-self:: **section** sélectionne tous les descendants **section** du nœud contextuel et le nœud contextuel lui-même si c'est une **section ;**
- self::**section** sélectionne le nœud contextuel si c'est un élément **section**, et rien dans le cas contraire ;
- child::**chapitre**/descendant:: **section** sélectionne les descendants **section** de l'élément **chapitre** enfant du nœud contextuel ;
- child::*/child::**paragra** sélectionne tous les descendants **paragra** du nœud contextuel ;
- /sélectionne la racine du document (qui est toujours le parent de l'élément document) ;
- /descendant::**section** sélectionne tous les éléments **section** descendants du document contenant le nœud contextuel ;
- /descendant::**section**/child::**avantages** sélectionne tous les éléments **avantages** qui ont un parent **section** et qui sont dans le même document que le nœud contextuel ;
- child::**section** [position()=1] sélectionne le premier enfant **section** du nœud contextuel ;
- child::**section** [position()=last()] sélectionne le dernier enfant **section** du nœud contextuel ;
- child::**section** [position()=last()-1] sélectionne l'avant dernier **section** enfant du nœud contextuel ;
- child::**section** [position()>1] sélectionne tous les enfants **section** du nœud contextuel autres que le premier ;
- /descendant::**image**[position()=42] sélectionne le 42ième élément **image** du document ;

- /child::doc/child::**chapitre**[position()=2]/child::section[position()=2] sélectionne la 2ième section du 2ième élément **chapitre** de l'élément **doc** du document ;
- child::**para**[attribute::type="**sous-section**"] sélectionne tous les enfants **para** du nœud contextuel qui ont un attribut type dont la valeur est sous-section **;**
- child::**chapitre**[child::title='Introduction'] sélectionne l'enfant **chapitre** du nœud contextuel qui a un ou plus enfant title avec un contenu textuel égal à 'Introduction' ;
- child::*[self::**chapitre** or self::appendix] sélectionne tous les enfants **chapitre** et **appendix** du nœud contextuel ;
- child::*[self::**chapitre** or self::**appendix**][position()=last()] sélectionne le dernier enfant **chapitre** ou **appendix** du nœud contextuel.

Dans Xpath il existe deux types de chemins de localisation : relatifs et absolus.

- **Un chemin relatif** consiste en une séquence d'une ou plusieurs étapes séparées par le caractère /. Les étapes sont assemblées de gauche à droite. Chacune à leur tour, les étapes sélectionnent un ensemble de nœuds relativement au nœud contextuel; c'est-à-dire, une séquence initiale d'étapes et composée avec une suivante comme suit : la séquence initiale sélectionne un certain nombre de nœuds qui sont utilisés un par un comme nœud contextuel de la prochaine étape. Les ensembles des nœuds identifiés par cette étape sont ensuite réunis.
- **Un chemin absolu** consiste en un / optionnellement suivi d'un chemin relatif. Un / par lui-même sélectionne le nœud racine du document contenant le nœud contextuel. S'il est suivi d'un chemin relatif, alors le chemin de localisation sélectionne l'ensemble des nœuds qui seraient sélectionnés par le chemin relatif s'appliquant au nœud racine du document contenant le nœud contextuel.

### 3.3.2 Étapes de localisation

Les chemins de localisation (*location paths*) sont constitués d'un ou plusieurs chemins de localisation séparés par le caractère /. Chaque chemin de localisation comprend trois parties distinctes :

1. Un **axe**, qui spécifie la nature des relations qui lient les différents nœuds d'un ensemble résultant. Il oriente le "location step" en lui donnant un sens de parcours dans l'arborescence du document xml entre le nœud contextuel et les nœuds à localiser. Les axes disponibles sont : child, preceding, descendant, attribute, parent, namespace, ancestor, etc.
2. Un **nœud de test**, précise le type des éléments que l'on souhaite inclure à partir des nœuds obtenus par l'étape de localisation.
3. **0 ou *n* prédicats**, qui sont des expressions arbitraires pour raffiner l'ensemble des nœuds obtenus par l'étape de localisation.

Du point de vue syntaxique, une position d'expression de chemin se représenterait de la façon suivante :

*axis::node-test [predicate]**

La syntaxe d'une étape de localisation est le nom de l'axe et le nœud de test séparés d'un point-virgule (:), suivi de 0 ou *n* expresions repérées par des crochets.

Par exemple, dans : `child::section[position()=1]`, **`child`** est le nom de l'axe, **`section`** le nœud de test et [`position()=1]`, `[child::title]`, sont des exemples des prédicats.

## 3.4 XPointer : le langage d'adressage de XML

XPath et XPointer sont deux spécifications de XML qui visent à améliorer les possibilités de définition des liens dans les documents XML. En bref, si XLink régit comment les liens sont insérés dans des documents XML, XPointer fournit la spécification de localisation à savoir, où le lien commence et où il finit exactement. XPointer est basé sur XPath qui définit comment adresser (ou repérer) les parties (fragments) d'un document XML. XPath et XPointer ont atteint le statut de la recommandation du consortium W3C [W3C-XPath], [W3C-XLink].

### 3.4.1 L'adressage avec XPointer

Le premier objectif de XPointer est de fournir un moyen simple d'adresser la structure interne d'un document XML (une partie). Pour cela, il utilise le langage XPath qui décrit des "patterns" (chemins de localisation) et extrait des valeurs de l'arbre du document XML. Grâce à XPointer, on peut créer un lien vers n'importe quel endroit d'une page, **sans avoir besoin d'ancre** comme pour HTML, et donc sans avoir besoin de modifier la page cible. XPointer permet entre autres de pointer vers **un ou plusieurs éléments** XML précis, vers un certain **texte**, ou vers **une partie** d'un document XML.

A la différence XPath opère sur des **nœuds** et des **ensembles de nœuds** à l'intérieur d'un document XML. XPointer dispose de fonctions de manipulations des localisations pour opérer sur :

- un point, (par exemple : localiser un **texte** à l'intérieur d'un élément)
- un nœud,
- une région (c'est-à-dire définir une portion entre deux points ou emplacements à l'intérieur d'un document) et des ensembles de celles-ci.

Pour cela, il fournit un ensemble de termes de localisation, dont les plus importants avec des exemples sont décrits à la suite.

La forme générale d'un lien se définit en ajoutant un XPointer à la fin d'une URI, après un caractère #, (URI#XPOINTER), où URI est un « *Universal Ressource Identifier* ». Ainsi, il devient très simple de combiner XLink et XPointer pour obtenir des référencements puissants. Ce qui a été dénoté dans le contexte de nœud de XPath est dénoté aussi dans le contexte XPointer.

### 3.4.2 Termes de localisation absolus avec l'attribut id ( point )

Ils permettent d'identifier la localisation source au départ du XPointer. Cette localisation source est en fait un élément d'un document XML. Le plus utilisé est le terme « **id()** » qui permet de sélectionner l'élément dans le document qui possède un attribut du type "ID" avec une valeur donnée.

La valeur de l'attribut « ID » étant unique et exclusive, le XPointer pointe sur un seul élément. Notons qu'il pointe **sur l'élément entier** et pas seulement sur sa balise de départ comme dans les liens HTML.

Dans XML, tout élément peut avoir un attribut id à valeur unique et exclusive, voyons l'exemple suivant :

```
<EleveINSA id="ALVAREZ" type="doctorant">
  ALVAREZ Abraham
</EleveINSA>
```

XML propose un lien "basique" avec l'attribut idref.

```
<EleveINSA>LIRIS<auteur idref="ALVAREZ"/></EleveINSA>.
```

**Ce chemin définit un point dans l'arborescence du document.** On peut l'identifier grâce à l'URL : http://lisi.insa-lyon.fr/alvarez.xml/#Xpointer(idref("ALVAREZ")).

Voyons un autre exemple : `Xpointer(id("3600"))`. Ici, le lien pourra adresser (pointer) vers l'élément "`SubFam`" possédant un attribut "`id`" de valeur "`3600`". La représentation arborescence du code xml pour cet exemple est la suivante :

```
<FamilleProduct id = "Samsung">
    .
    .
<FamilleProduct id="Nokia">
   <SubFam id ="3600"> <-------- Xpointer (id("3600"))
      <modele>3650</modele>
      <poid>190 grammes</poid>
      <Communication>Tribande</Communication>
      <Prix>650.00</Prix>
   </SubFam>
  <SubFam id ="P800">
      <modele>S55</modele>
      <Prix>480.00</Prix>
  </SubFam>
</FamilleProduct>
```

Les liens utilisant le terme "`id()`" sont les plus robustes vis-à-vis de modifications des documents et sont les plus conseillés dans la mesure du possible. De plus, ce terme permet la compatibilité avec les ancres HTML existantes : "`#intro`" en HTML correspond à "`#xpointer(id("intro"))`". Il existe d'autres termes de ce

type, comme "`root()`" représentant la racine d'un document XML, ou "`here()`" permettant de localiser l'élément lien servant d'origine dans le cas d'un lien interne à un document. Les termes de localisation absolus peuvent être suivis de termes de localisation relatifs.

### 3.4.3 Termes de localisation relatifs

Ils définissent la direction à suivre par rapport à une localisation source qui peut être soit le terme de localisation absolu, soit le terme de localisation relatif précédent. Il existe douze termes relatifs, permettant par exemple de se diriger vers les éléments enfants, descendants, parents ou ancêtres de l'élément servant de localisation source.

Considérons le terme "`child`" qui indique les enfants d'un élément. Dans l'exemple suivant, le XPointer pointe vers les deux éléments "`i800`" qui sont des enfants de l'élément "`NokiaSubFam`" dont l'attribut "ID" a la valeur "`800Wap`" :

```
<FamilleProduct id = "Samsung">
     .
     .
<FamilleProduct id= "Nokia">
  <SubFam id ="3600">
      <modele>3650</modele>
      <poid>190 grammes</poid>
      <Communication>Tribande</Communication>
      <Camera Integre>oui</Camera Integre>
      <Prix>650.00</Prix>
  </SubFam>
  <NokiaSubFam id ="800Wap">  ◄----  Xpointer (id("800Wap"))/child::i800
      <i800> ◄..........................  element 1
         <modele>P800W</modele>
         <WapVersion>2.3</WapVersion>
         <Navigateur>non</Navigateur >
      </i800>
      <i800> ◄..........................  élément 2
         <modele>P800i</modele>
         <WapVersion>2.0</WapVersion>
         <Navigateur>non</Navigateur >
      </i800>
</FamilleProduct>
```

Parmi les autres termes de ce type, on peut remarquer aussi le terme "attribut" qui localise un ou plusieurs éléments selon la valeur d'un de leurs attributs. XPointer permet de créer un lien vers n'importe quel endroit d'une page, sans avoir besoin d'`id`:

```
XPointer(EleveINSA[text() = "ALVAREZ Abraham"])
```

Ici l'élément `EleveINSA` dont le texte est ALVAREZ Abraham. **Ce chemin définit un nœud dans l'arborescence du document**.

### 3.4.4 Termes de localisation de parties (une portion)

XPointer permet de pointer vers plusieurs éléments XML précis, un texte, ou encore vers une partie d'un document XML. Les termes de localisation de parties permettent de localiser toute une partie d'un document XML située entre deux emplacements dans le document appelés "points" ou "pointeurs". On utilise dans la syntaxe de Xpointer le terme "`to`" entre deux suites de termes définissant chacun un point dans un même document.

Par exemple, sélectionner tous les renseignements dans l'élément `SubFam` depuis l'attribut modele ="`Wap3650`" jusqu'à l'attribut WapVersion="`2.0`" :

**Xpointer(Wap3600/@ modele="3650" to Wap3600/@WapVersion="2.0")**

```
<FamilleProduct id = "Samsung">
          .
          .
<FamilleProduct id="Nokia">
  <SubFam id ="Wap3600">
      <modele>3650</modele>
      <poid>190 grammes</poid>
      <Com>Tribande</Com>
      <Prix>650.00</Prix>
      <Navigateur>non</Navigateur >
      <WapVersion>2.0</WapVersion>
      <Camera Integre>oui</Camera Integre>
  </SubFam>
</FamilleProduct>
```

**portion sélectionnée** (lignes de `<modele>` à `<WapVersion>`)

**Ce chemin définit une région entre 2 points dans l'arborescence du document.**

### 3.4.5 Termes de localisation de chaînes

XPointer dispose de fonctions de manipulations des localisations (point, nœud, région) et des ensembles de celles-ci. Les termes de localisation de chaînes permettent de localiser un texte dans un document XML. On peut utiliser le terme "string-range()":

XPointer(string-range(publication[auteur/@idref="alvarez"], "XML")[position()=last])

Ici, Xpointer sélectionne la dernière position des localisations de la chaîne "XML" dans les publications de l'auteur Alvarez.

Voyons l'exemple suivant où le XPointer pointe vers la deuxième occurrence de la chaîne "Tribande" dans l'élément avec la valeur id="Motorola".

```
<FamilleProduct id = "Nokia">
          .
<FamilleProduct id = "Motorola">
  <MotoFam id = "3600">
      <modele>3650</modele>
      <poid>190 grammes</poid>
      <Autonomie>04h</Autonomie>
      <Comm>Tribande</Comm> <------- Première occurrence
      <Prix>650.00</Prix>
  </MotoFam>
  <SiemensFam id = "Marlin">
      <modele>S55</modele>
      <poid>160 grammes</poid>
      <Autonomie>06h</Autonomie>
      <Comm>Tribande</Comm <------- Deuxième occurrence
      <Prix>680.00</Prix>
  </SiemensFam id>
  <SonySubFam id = "800Wap">
      <i800>
      <modele>P800W</modele>
      <WapVersion>2.0</ WapVersion>
      <NavigateurWeb>OUI</ NavigateurWeb>
      </i800>
  </SonySubFam>
</FamilleProduct>
```

**Xpointer (string-range id("Motorola"),**

Nous avons vu que XPointer permet un repérage des éléments et du texte avec une granularité fine, alors que dans HTML, il est nécessaire de fournir une

"identificateur" du type (*id-valeur*) pour chaque balise que l'on veut répertorier. De plus, XPointer offre la possibilité de localiser des éléments sans utiliser une "*id-valeur*". Ceci est particulièrement utile si l'auteur d'un document ne veut pas tenir compte du fait que les parties du document seront des ancres destination des liens.

# 4 Autre catégorie de liens

Randall propose une catégorisation de 80 types de liens [TW83], on constate qu'une trop grande variété de types de liens transforme la désorientation de l'utilisateur de l'hypertexte en confusion. Cette classification nous a inspiré dans la classification que nous proposons ci-dessous (C.f. Sections 4.1, 4.2 et 4.3). Dans le cadre de nos recherches, nous nous proposons de construire une classification des liens en fonction des opérations à effectuer, dans le but de pouvoir les contrôler et les manipuler. On peut les trouver sous diverses formes, ils peuvent s'imbriquer comme un point de départ à destination d'une page, entre deux sections, ou une note de bas dans un document ou ceux-ci peuvent aussi représenter la liaison de structure entre un paragraphe dans un document.
La détermination automatique des types de liens est notamment proposée par James Allan [Allan95]. D'après James Allan [Allan96], [W3C-XLink] un type de lien est défini comme :

> *« Une description de la relation entre la source et la destination d'un lien »*

## 4.1 Liens de service

### ❖ Lien de navigation (notion transversale)

Une des raisons principales du succès du Web, est le concept « pointer et cliquer ». Ce concept est le fondement de l'hypertexte, il est le plus populaire dans les systèmes d'hypertexte et hypermédia. Ces liens sont de type implicite.

### ❖ Lien linguistique

C'est un type de lien que l'on peut trouver facilement lorsque par exemple on utilise la technique simple de recherche par « *Pattern Matching* ». Un exemple, consiste en la mise en correspondance des mots d'un document avec un dictionnaire. Dans la plupart des cas, ces liens ont pour source un mot ou une phrase d'un document [Bernstein98].

### ❖ Liens d'information (aussi nommés « *voir aussi* »)

Les liens ont fréquemment pour fonction de souligner l'existence davantage d'informations sur le sujet en cours de visualisation. Ce sont des «liens de référence» [Thistlewaite95], par exemple : « *pour plus de détails cliquer ici* ». Ces liens qui renvoient à d'autres pages ou paragraphes du texte, constituent également des liaisons directes vers des unités implicites de sens, localisées au sein d'une page ou d'un paragraphe. Voir le concept de notion transversale (C.f. Section 4.1).

## 4.2 Liens de structure

### ❖ Liens structurels

Les liens structurels sont ceux qui retracent le schéma ou la structure logique d'un document. Par exemple, sont autant des liens structuraux les liens pour naviguer entre chapitres ou sections, les liens qui désignent une figure, les liens qui pointent vers une citation bibliographique (voir l'article …) lors de la citation des travaux d'un auteur dans une bibliographie.

### ❖ Liens de récupération

La première étape d'interaction entre l'utilisateur et le système, lorsqu'un document d'URL donnée est récupéré depuis le serveur, peut être considérée comme un lien. Dans le cas des documents structurés, ils représentent la structure logique du document (DOM – *Document Object Model*). DOM permet de naviguer, de gérer son contenu, d'accéder à la structure et au style, en définissant un ensemble de classes, indépendamment des plats-formes et des langages de programmation [Wood99], [W3C-DOM]. L'accès aux données est réalisé par des nœuds qui les contiennent.

❖ **Liens de parcours**

Ces liens entre les documents offrent une fonction ergonomique de lecture rapide dans une collection de documents.

❖ **Liens de citation**

Ce type de lien est utilisé généralement dans le but d'offrir un contexte additionnel. Par exemple, en offrant la possibilité de cliquer sur des références contenues dans des articles de recherche, dans une base de données comme la base du site CiteSeer [GBL98] – *Scientifique Literature Digital Library* (*ResearchIndex*), qui actuellement contient plus de 500.000 documents selon Lawrence [LGB99]. Ce même Lawrence qui a par la suite essayé de délocaliser de la base les adresses URL non valides et discute l'utilisation des URL dits persistantes [LCGF++2000]. Le projet « Open Journal » s'appuie sur ce type de lien dans le but d'élargir la diffusion des journaux sur le Web [HCHHH97].

## 4.3 Liens de fonction

❖ **Représenter des actions**

Un lien de fonction peut par exemple avoir pour vocation de représenter une action. L'action est lancée après l'activation du lien. Cela peut être : *ouvrir une autre page*, *activer une applet* ou *un script java*, etc. Dans le passé, le téléchargement et la recherche d'information sur des sites distants ont pu être problématiques pour certains utilisateurs, car cette tâche demandait un certain savoir-faire comme par exemple pour télécharger un fichier par ftp, ou comme pour utiliser des outils comme Gopher. Les browsers Web, depuis Mosaic [NCSA95], offrent des liens hypertextuels pour réaliser cette fonction à l'aide d'une interface graphique.

❖ **Liens associés aux protocoles**

Cette catégorie de liens est basée sur les types de documents liés entre eux. Nous avons inclus cette catégorie parce qu'ils sont typiquement identifiés par une « ressource » vers laquelle ils se connectent. HTTP, FTP, TELNET, SMPT, WAIS, et tout dernièrement les protocoles SOAP [Englander03], [W3C-SOAP], et WSDL

[W3C-WSDL]. Ils sont identifiés par une « balise » ou code dans le texte du lien. i.e. **http**://www.google.fr/, **ftp**://ftp.lisi.insa-lyon.fr.

# 5 Conclusion

La spécification XPointer n'est pas conçu pour être un langage d'interrogation. Parfois, il est trop compliqué de définir un chemin d'accès si on ne connaît pas d'une façon générale la structure hiérarchique du document XML.

Nous avons vu, que par l'intermédiaire de XLink et/ou XPointer, le mécanisme de liens XML peut donc lier des documents XML avec de nombreuses possibilités. Une fonction très utilisée est la fonction `id()` : il est défini d'ailleurs, dans la spécification de Xpath, et cette fonction choisit un élément dans un document XML qui a un attribut de type `id` avec une valeur spécifiée (C.f. Section 3.3.2) [W3C-XPath] pour une définition exacte des paramètres. L'inconvénient de la fonction `id()` est qu'elle peut seulement être utilisée pour des documents qui ont a un attribut `id` défini. Si aucun attribut du type `id` n'est présent et si n'existe aucune manière de modifier le document cible, alors que le chemin de localisation doit être utilisé.

En outre, retenons que le mécanisme de liens peut donc lier des documents XML avec de nombreuses possibilités et surtout qu'il surpasse les liens HTML pour les raisons suivantes :

- n'importe quel élément XML peut être un lien.
- XLink peut préciser le comportement des applications par rapport aux liens, notamment ouvrir un lien dans une nouvelle fenêtre.
- XLink permet des liens entre plus de deux ressources, bi-directionnels, multi-directionnels et externes aux documents liés.
- les liens hors ligne peuvent lier des ressources en lecture seule ou inaccessibles, et permettent ainsi une mise à jour plus souple des liens.
- XPointer offre de nombreuses possibilités pour localiser un endroit à l'intérieur d'un document XML, et même des parties entières d'un document.

## 5.1 Résumé de la topologie des liens

Afin d'avoir un panorama global des liens abordés dans ce chapitre, nous présentons une synthèse graphique.

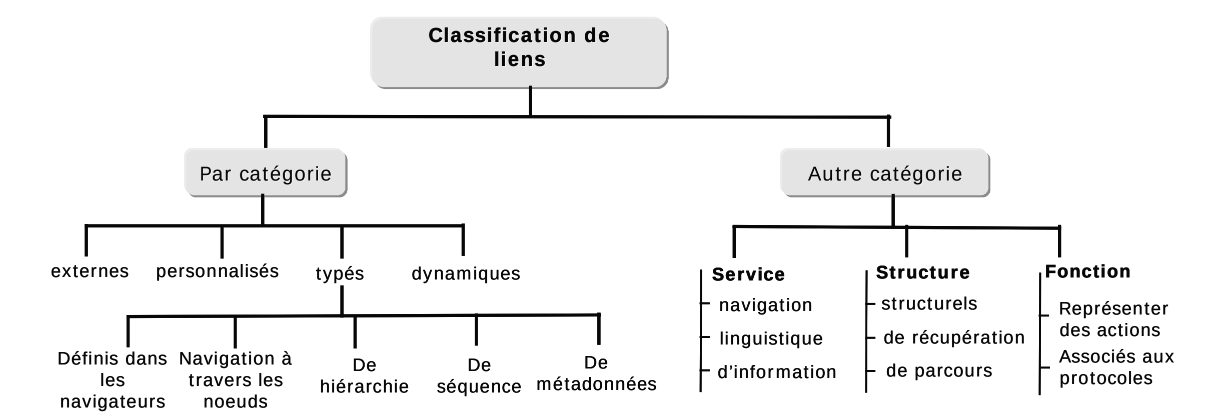



*Figure 1-8 Représentation graphique de la classification des liens*